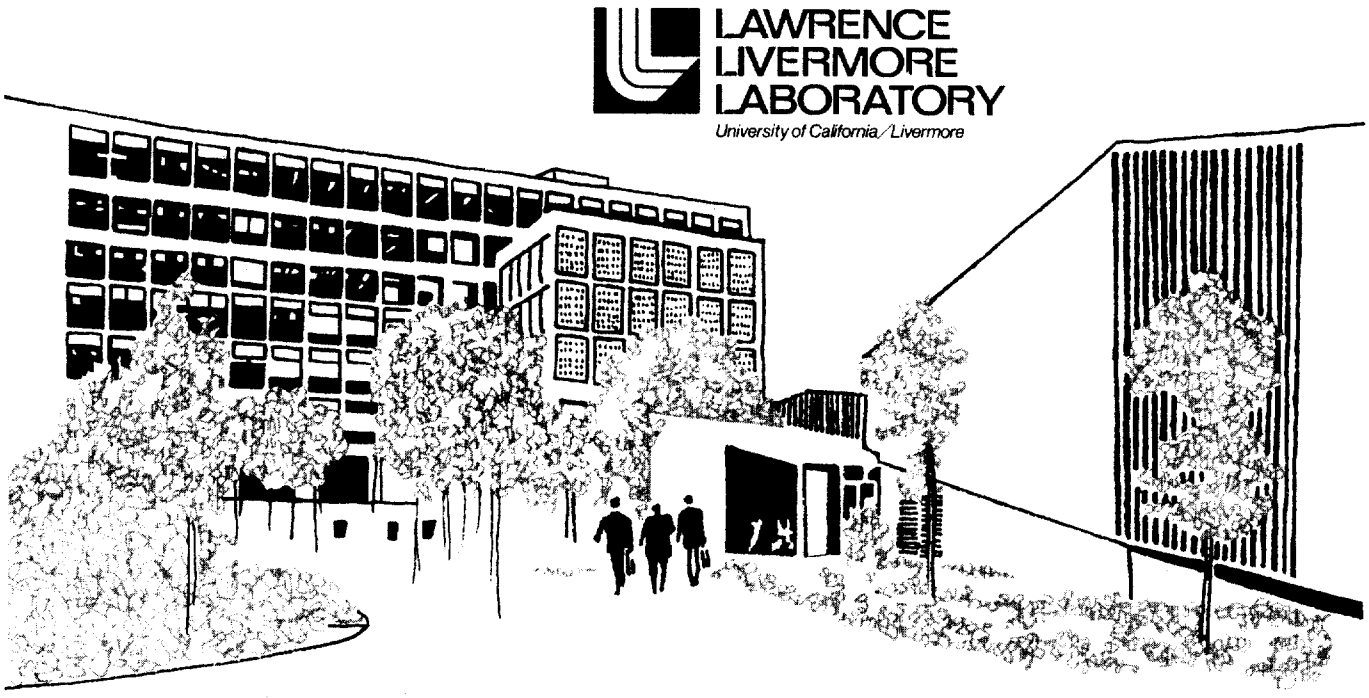
UCRL-52609

USER'S MANUAL FOR PELE-IC: A COMPUTER CODE FOR EULERIAN HYDRODYNAMICS

W. H. McMaster
E. Y. Gong

May 29, 1979

Work performed under the auspices of the U.S. Department of Energy by the UCLLL under contract number W-7405-ENG-48.



Work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract W-7405-ENG-48.

Distribution Category
UC-32

LAWRENCE LIVERMORE LABORATORY
University of California/Livermore, California/94550

UCRL-52609

USER'S MANUAL FOR PELE-IC: A COMPUTER CODE FOR EULERIAN HYDRODYNAMICS

W. H. McMaster
E. Y. Gong

Manuscript date: May 29, 1979

Contents

USER'S MANUAL FOR PELE-IC:
A COMPUTER CODE FOR EULERIAN HYDRODYNAMICS

ABSTRACT

The PELE-IC code is a two-dimensional semi-implicit Eulerian hydrodynamics computer program for the solution of incompressible flow coupled to flexible structures. The fluid, structure, and coupling algorithms have been verified by the calculation of solved problems from the literature and from air and steam blowdown experiments. The code is written in both plane and cylindrical coordinates using the following set of units; pressure in bars, distance in cm, and time in ms. The coupling algorithm is general enough to handle a wide variety of structural shapes. This user's manual gives a detailed explanation of the algorithms to help potential users to understand the capabilities and limitation of the code.

1.0 INTRODUCTION

The basic semi-implicit solution algorithm contained in the SOLA code[1] was used as a foundation for the development of the PELE-IC code. In the PELE-IC code we use the concepts of void fractions and interface orientation to track the movement of free surfaces. This gives us great versatility in following fluid-gas interfaces both for bubble definition and water surface motion without the use of marker particles.

The structural motion is computed by a finite element code[2] from the applied pressure at the fluid-structure interface. The shell structure algorithm uses conventional thin-shell theory with transverse shear. The finite-element spacial discretization employs piecewise-linear interpolation functions and one-point quadrature applied to conical frustra. We use the Newmark implicit time integration method implemented as a one-step module. The fluid code then uses the structure's position and velocity as boundary conditions. The fluid pressure field and the structure's response are corrected iteratively until the normal velocities of the fluid and the structure are equal. This results in a strong coupling between the two algorithms.

A simplified theory of condensation on a free surface due to Kowalchuk and Sonin[3] has been included to calculate the effects of steam condensation

and oscillatory chugging on structures. Other models can readily be tested in the PELE-IC code.

The physics of the code and some illustrative problems used to verify the solution algorithms are given in UCRL-52620.[4]

2.0 THE PELE-IC SOLUTION ALGORITHMS

2.1 INTRODUCTION

The underlying approach used by PELE-IC for the solution of general flow fields is the use of the semi-implicit SOLA solution algorithm. The basic assumption of this approach is that all flow within the computational grid is incompressible. Superimposed on this basic algorithm we have applied the boundary conditions for free surfaces, confined compressible gases, and moving structures.

The solution strategy is to first solve the Navier-Stokes equations explicitly using values from the previous time step. This yields trial values for the flow field $(\tilde{U},\tilde{V})$, where the tilde denotes that these are tentative velocities. Since these velocities do not necessarily satisfy the continuity equation, we iterate on the pressure field until the incompressibility condition for each computational cell is satisfied within prescribed limits. During the iteration, the boundary conditions imposed by free surfaces and structures are applied and the iteration is continued until both the incompressibility and boundary conditions are satisfied. This, then yields new values for the field variables (P,U,V) at the new time level $t+\delta t$. These new values are then used to locate all free surfaces and move the marker particles which are used to visualize the flow field.

2.2 SOLA SOLUTION ALGORITHM

The equations of motion are the Navier-Stokes equations:

$$\frac{\partial u}{\partial t} + \frac{\partial u^2}{\partial x} + \frac{\partial uv}{\partial y} + \frac{\xi u^2}{x} = - \frac{\partial p}{\partial x} + \nu \left[\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + \xi \left(\frac{1}{x} \frac{\partial u}{\partial x} - \frac{u}{x^2} \right) \right]$$

$$\frac{\partial v}{\partial t} + \frac{\partial uv}{\partial x} + \frac{\partial v^2}{\partial y} + \frac{\xi uv}{x} = - \frac{\partial p}{\partial y} + g_y + \nu \left[\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} + \frac{\xi}{x} \frac{\partial v}{\partial x} \right] . \qquad (1)$$

The solution of these equations is accomplished using the following strategy. We just rewrite the Navier-Stokes equation in terms of time level as

$$\frac{\partial u}{\partial t} = \left[-\nabla\, uu + g + \nu\nabla u^2\right]^n - \nabla p^{n+1} \quad . \tag{2a}$$

If we set $p^{n+1} = p^n + \delta p$ then the equation becomes

$$U^{n+1} = \left\{u + \delta t\left[-\nabla\cdot uu - \nabla p + g + \nu\nabla u^2\right]\right\}^n - \delta t\, \nabla\, \delta p$$

or

$$u^{n+1} = \tilde{u} - \delta t\, \nabla\delta p \quad , \tag{2b}$$

where we have defined the terms in the braces as $\tilde{u}$. By an iteration process we solve for $\nabla\delta p$ such that the continuity equation

$$\nabla\, u^{n+1} = 0 \tag{3}$$

is satisfied. To do the iteration we define for each iteration, i, $\nabla\cdot u_i = D_i$ and choose the first trial velocity of the iteration as $u_1 = \tilde{u}$. The final pressure for a cell is then found from a Newton-Raphson iteration

$$\delta p = \sum_i \delta p_i \quad ,$$

where

$$\delta p_i = \frac{-D_{i-1}}{\partial D/\partial p} \quad . \tag{4}$$

Thus, in this algorithm, the incompressibility constraint is imposed by adjusting the cell pressure with a resulting change in the flow field. For example, if the divergence of a cell is negative corresponding to a net flow of mass into the cell, the pressure is increased to eliminate the inflow. Likewise, when there is a net flow out of the cell, the pressure is decreased to draw it back. Since the pressure is cell centered and the velocities are on the cell sides, the divergence can be driven to zero in this manner.

The pressure adjustment must be done iteratively; however, because when one cell is adjusted, the velocities of its neighbors are also affected. The iteration proceeds by sweeping the mesh rows from left to right starting with the bottom row and working upward. For each cell encountered, the divergence, D, is computed using the most current velocity values available. The pressure change, δp, required to make D = 0 is then determined. The iteration proceeds until all computational cells satisfy the criteria that $D \leq \epsilon$, where ϵ is a small number specified by the user.

The pressure changes determined in this way are calculated from the formula:

General Case

$$\delta p_i = \frac{-D_{i-1}}{2\,\delta t\left(\frac{1}{\delta x^2} + \frac{1}{2\delta y^2}\right)} \tag{5}$$

Rigid Boundary on One Side

The geometric factor in Eq. (5) is altered if the cell is bounded by a rigid boundary since, in this case, there is a reduction in the number of degrees of freedom; i.e., there are fewer velocity components we are free to adjust.

As an example, consider the case where the bottom boundary of the cell is rigid and it's normal velocity is zero for all time; i.e., V8 = 0. Then we have

$$\delta p = \frac{-D_{i-1}}{2\,\delta t\left(\frac{1}{\delta x^2} + \frac{1}{2\delta y^2}\right)} .$$

Similarly for a rigid boundary on a side of the cell

$$\delta p_i = \frac{-D_{i-1}}{2\,\delta t\left(\frac{1}{2\delta x^2} + \frac{1}{\delta y^2}\right)} .$$

Rigid Boundary on Two Adjacent Sides

For the corner cell such as one on axis with U6 = V8 = 0, then we have

$$\delta p_i = \frac{-D_{i-1}}{\delta t\left(\frac{1}{\delta x^2} + \frac{1}{\delta y^2} + \frac{\xi}{2x\delta x}\right)} \quad .$$

In practice we multiply δp by an over-relaxation parameter, ω, to help accelerate the convergence.[5] For stability ω has the restriction $0 \leq \omega < 2$.

Since we are computing the new pressure field at a cell center based on the cell velocity divergence, we are not required to apply explicit pressure boundary conditions at structural interfaces.

2.3 FREE SURFACE ALGORITHM

Accurate free surface tracking is necessary to allow the application of appropriate velocity and pressure boundary conditions. In our approach, we use the concepts of void fractions and surface orientation to define the free surface. This method automatically guarantees mass conservation in the surface cells when we do the advection.

The geometry considered by the free surface algorithm is shown in Fig. 1 where we have shown an arbitrarily oriented interface.

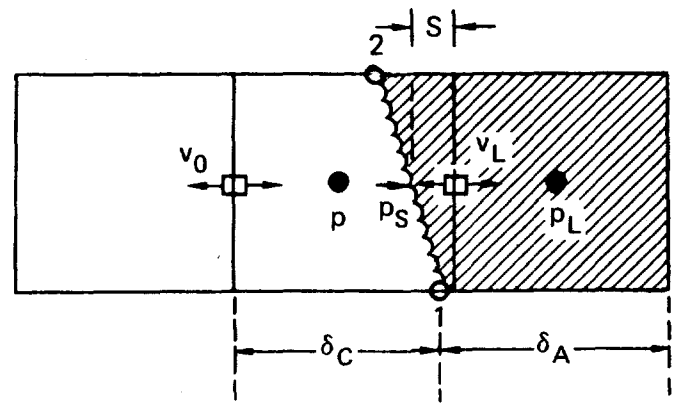


FIG. 1. Geometric considerations for the free surface algorithm.

The free surface algorithm performs four functions:

1. Applies the prescribed boundary pressure,
2. determines the surface orientation,
3. calculates the fluid advection, and
4. sets the void velocities to maintain cell incompressibility.

Boundary Pressure

The cell centered pressure, p, to be used in the equations of motion is determined from a linear interpolation or extrapolation based on the prescribed boundary surface pressure, P_S, and the current fluid pressure, P_L, in the adjacent liquid cell. If there is more than one adjacent full cell, then the average of all the interpolated values is used. The surface boundary pressure can either be a constant or a prescribed function of time. Thus,

$$P = P_L + (P_S - P_L)(\delta A + S)/(\delta A + \delta C) \quad ,$$

where the cell widths are given by δA and δC, and the surface location is given by $S = 1 - \theta$, with θ being the void fraction of the cell.

Surface Orientation

The surface orientation is specified by its intercepts on two sides of the cell as illustrated by points 1 and 2 in Fig. 1. These points are determined by the void fractions of the cell and its four nearest neighbors. Within the cell, the interface is considered to be a straight line segment. Thus the surface is tracked by its intersection of grid lines.

The cell side intercept of a free surface depends upon the relative void fractions in each of the two mixed cells with a common side. The code distinguishes the four cases shown in Fig. 2.

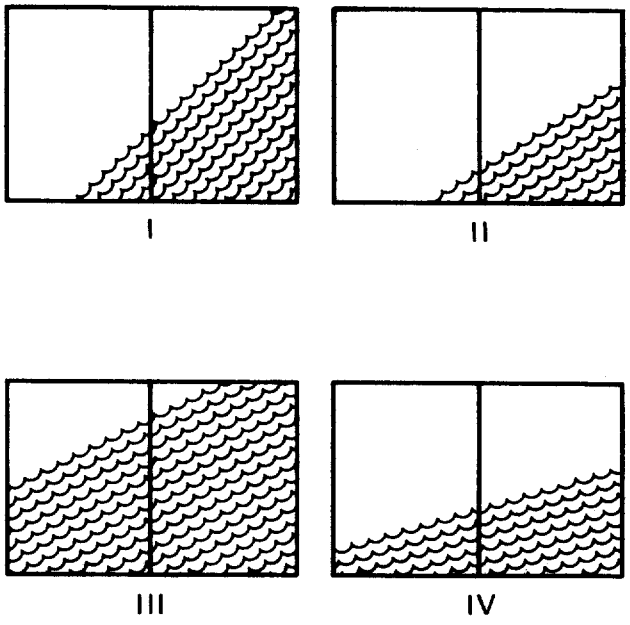


FIG. 2. Possible surface orientations between cells C and A.

Figure 2 shows the case for an intercept on side 1 of cell C and side 3 of cell A. Intercepts on other sides are just rotations of this basic orientation and their intercepts are found in the same manner.

In the solution algorithm, we indicate the larger of the two void fractions as θ_L and the smaller as θ_S. The cell widths associated with these void fractions are X_L and X_S, respectively, and we define their ratio as

$$R = X_L/X_S \ .$$

Once the intercept is found we store the fraction of the cell side as measured in the positive direction. These fractions are contained in the cell identifier word CI(K) as MA5-MA8.

For finding the intercept, f, we make the assumption that the surface is a straight line segment in the two adjacent cells. This will give an approximate location which will be corrected for mass conservation when the advection is calculated.

The formulas used to find the cell side intercepts are found from simple geometric considerations with the following results:

Case I

$$f = \frac{\sqrt{R(1 - \theta_L)/\theta_S}}{1 + \sqrt{R(1 - \theta_L)/\theta_S}} \quad .$$

Case II

$$f = 2\left\{\sqrt{R(1 - \theta_L)\left[R(1 - \theta_L) + (1 - \theta_S)\right]} - R(1 - \theta_L)\right\} .$$

Case III

This case is just the inverse of Case II.

$$f = 1 - 2\left[\sqrt{\frac{\theta s}{R}\left(\frac{\theta s}{R} + \theta_L\right)} - \frac{\theta s}{R}\right] .$$

Case IV

$$f = \frac{R\theta_s + \theta_L}{R + 1} \quad .$$

Fluid Advection

The fluid is advected across cell boundaries using the donor cell method where the amount of liquid advected is determined from the contents of the upstream cell. The amount of donor cell liquid advected is determined from the fluid-void interface. This method provides an accurate tracking of the free surface without the requirement of using marker particles to define surfaces.

When the advection is across a side designated as mixed, then the advection is determined from either a trapezoidal or triangular region. The amount of liquid in the cell is given by $(1 - \theta)$ and we compute the fraction, F, of this amount that is advected from the one cell to the next.

(1) Trapezoid region

The geometry of the trapezoidal region is as shown in Fig. 3.

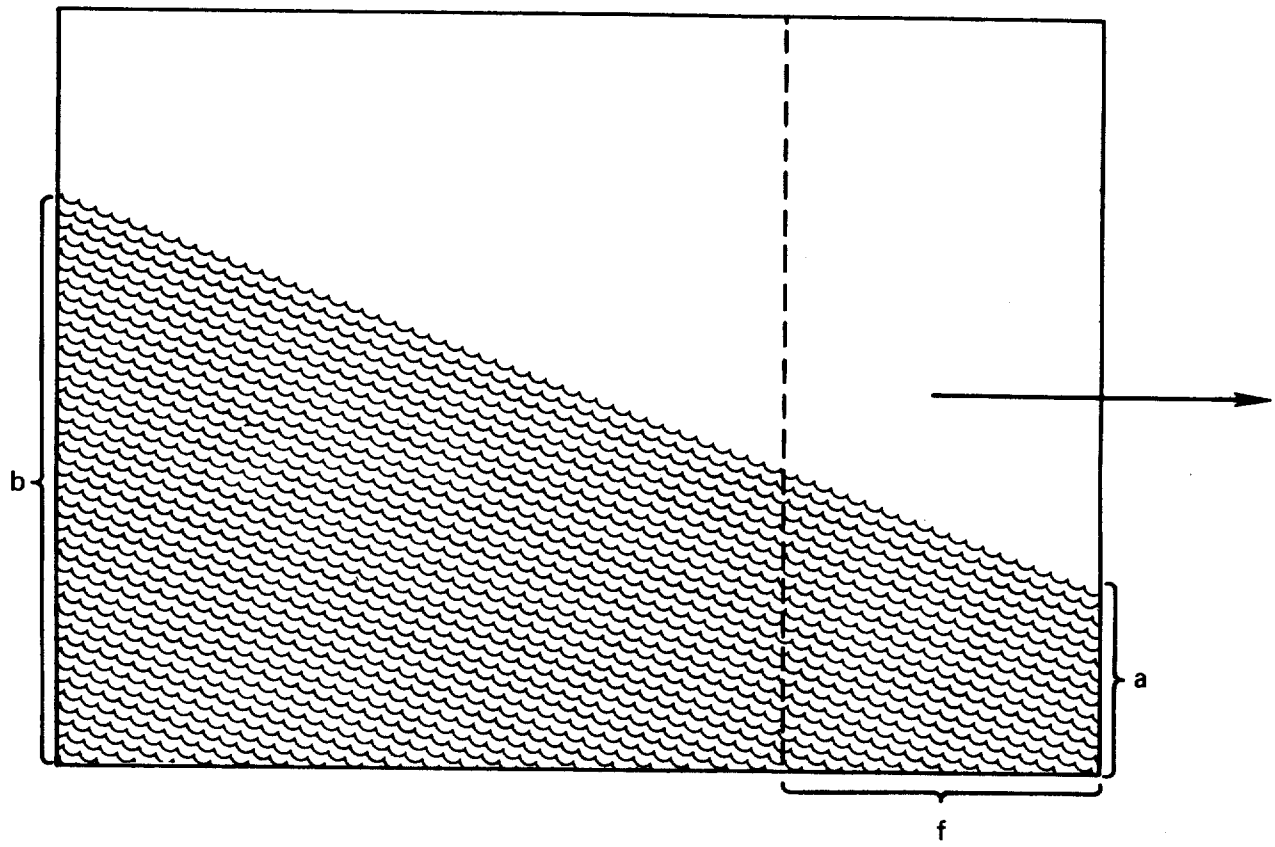


FIG. 3. Trapezoidal region.

From geometrical considerations, the fraction of the total liquid advected is given by

$$F = \frac{[a(2 - f) + bf]f}{(a + b)} \quad ,$$

where

$$f = \frac{u\delta t}{\delta x} \quad .$$

(2) Triangular region

The geometry of the triangular region is shown in Fig. 4.

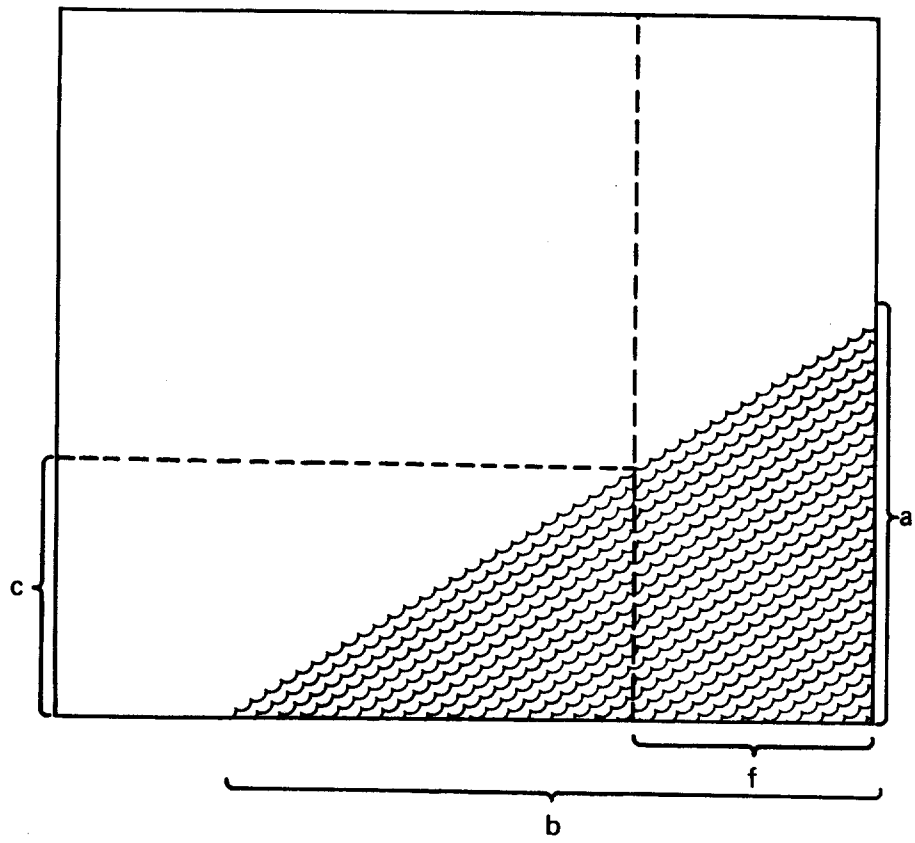


FIG. 4. Triangular region.

From geometrical considerations, the fraction of the total liquid advected is given by

$$F = \frac{(a + c)f}{ab} .$$

For a triangular liquid section. If the triangular section is void, then the fraction of the liquid transferred is

$$F = - \frac{(a + c)f\theta}{ab(1 - \theta)} .$$

Cell Incompressibility

The incompressibility of the cell is maintained by setting the void side velocity, VO, such that $\nabla^* V = 0$. This velocity is used both in the solution of the Navier-Stokes equations and to maintain continuity in the flow field when the surface motion causes the interface to cross grid lines.

2.4 FLUID-STRUCTURE INTERFACE ALGORITHM

This algorithm couples the fluid motion to the structure's motion by requiring that the normal velocities be equal at the interface and that the computational cell remain divergentless. This is accomplished in the iteration process by the adjustment of the pressure field.

The algorithm applies equally well to both rigid and moving surfaces. For rigid surfaces, the normal velocity is set to zero. For moving surfaces, the finite element module uses the pressure field supplied by the fluid and provides the fluid code with the resultant position and velocity of the interface. The interface coupling is accomplished at the intersection of an I or J line with the structure where an I line is defined as a vertical line through the cell center and a J line is a horizontal line through the cell center. The choice of I or J line coupling is determined from the angle the structure makes at the intersection. For angles of 45 degrees or less, we use I line coupling and J line coupling for larger angles.

The pressure applied to an element is determined from the average of all cells which contain nodes, I or J line intercepts as shown in Fig. 5. This average is weighed by the liquid content of the cell so that the proper pressure will be applied when a surface is in the same cell. Thus, we have

$$P = \frac{\sum_k (1 - \theta_k) P_k}{\sum_k (1 - \theta_k)} ,$$

where the k are the cells containing either nodal end points or intercepts.

The fluid and structure normal velocities are coupled at the I or J line intercepts. The normal fluid velocity is found by interpolation between all four of the cell side velocities and the structure velocity is found by interpolation between nodal values. We then use the structure's normal velocity to set the I or J cell side velocity which lies outside the structure; any other cell side velocities outside the structure are slaved to the nearest inside cell. These new velocities then cause the cell not to satisfy the divergence criteria and the pressure must then be adjusted using the SOLA algorithm. This change in the pressure field causes a different

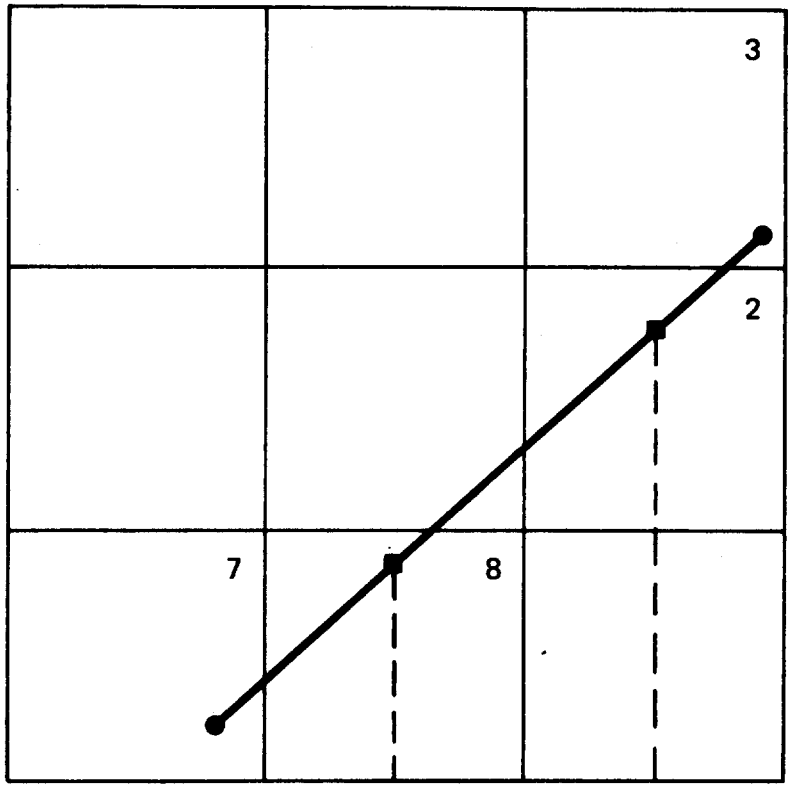


FIG. 5. Pressure averaging for application to a structural element. Cells 2, 3, 7, and 8 contribute to the pressure load.

structural response; therefore, the iteration proceeds until both conditions are satisfied. Within a single interation, all Eulerian zones are adjusted one by one, and then all the shell nodes are simultaneously adjusted by the implicit step.

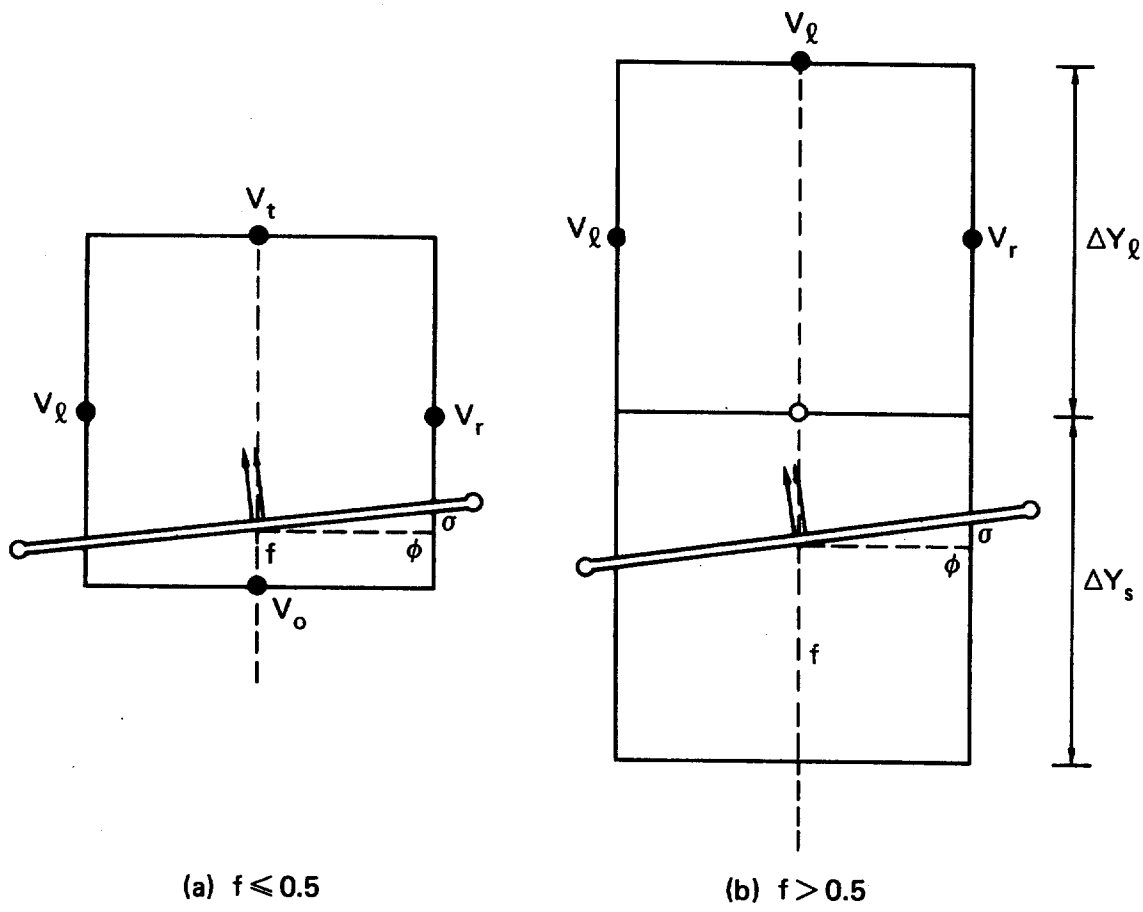
The fluid velocities are specified at cell sides and the structure velocities are specified at nodes as shown in Fig. 6a. The normal velocity of the structure at the intersection is given by

$$VS = \cos\phi \ (-U_S \tan\phi + V_S) \quad ,$$

while that of the fluid is

$$VF = \cos\phi \ (-\bar{U} \tan\phi + V_i) \quad ,$$

where ϕ is the angular orientation of the structure, $\bar{U} = 0.5(U_r + U_\ell)$ is the interpolated x-velocity along the I line, and V_i is the interpolated



(a) $f \leqslant 0.5$ (b) $f > 0.5$

FIG. 6. Geometry for the fluid-structure coupling algorithm. (a) Unblended cell couples velocity V_0. (b) Blended cell couples velocity V_t.

vertical fluid velocity at the I line intersection. These two velocities must be equal which gives us the basic coupling equation

$$V_i = V_s + \tan\phi\ (\bar{U} - U_s) \quad .$$

Using this equation we can determine the cell velocity outside the fluid as

$$V_0 = V_t + \left[(V_S - V_t) + \tan\phi\ (\bar{U} - U_S)\right] /(1 - F) \quad .$$

When F, approaches 0, then V_0 approaches V_S showing the strong coupling. However, as F approaches 1, V_0 becomes very large and the coupling to the structure becomes very weak. To avoid this condition, we blend cells whenever F is greater than 0.5.

In the blended case (see Fig. 6b), the velocity coupled to the finite element is V_t instead of V_0. We find the coupled cell side velocity from an interpolation between V_t and V_ρ with the result

$$V_t = V_\ell + \frac{DY_\ell}{DY_\ell + (1 - F)DY_S} \left[(V_S - V_L) + \tan\phi\ (\bar{U} - U_S)\right],$$

and we note that as F approaches 1, then V_T approaches V_S which gives us the strong coupling desired.

By performing this blending, we avoid discontinuities as the structure crosses a grid line as well as maintaining a strong coupling regardless of the structure's location in the fluid cell.

To see how the merging works, consider the situation shown in Fig. 7 where we have I line coupling and the structure moves down across the grid line. For simplicity we will assume equally spaced grid lines.

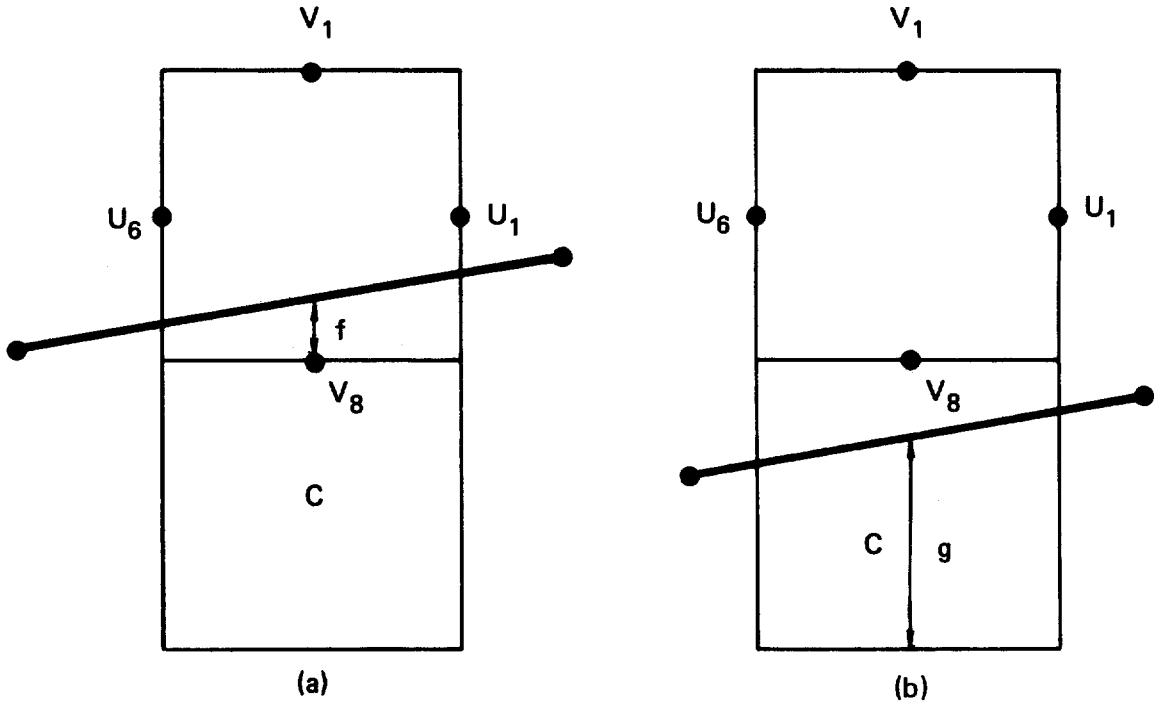


FIG. 7. Coupling as a structure moves across a grid line.

The coupling equation for Fig. 7a when the structure is in cell A is

$$V_8 = V_1 + \left[(V_s - V_1) + \tan\phi\ (U_A - U_S)\right] / (1 - f) \quad .$$

The coupling equation when the structure is in cell C, Fig. 7b is

$$V_8 = V_1 + \left[(V_s - V_1) + \tan\phi\ (\bar{U}_c - U_s)\right] / \left[1 + (1 - g)\right] \quad .$$

The g and f values are related by $g - 1 = f$ when $g \geq 1$, and upon substitution, we see that the two equations are identical if $\bar{U}_A = \bar{U}_C$.

In order for the coupling algorithm to work properly, every structural element must have at least one I or J intercept. Otherwise, even though the element is coupled to the fluid through the pressure, the fluid will not be coupled to the motion of the element.

3.0 STRUCTURE OF THE CODE

3.1 FINITE DIFFERENCE MESH

The finite difference mesh used for the numerical solution of the governing equations consists of rectangular cells of variable width δx and height δy. The generator allows one to increase or decrease the spacing geometrically to provide a smooth transition between all sizes. The grid can consist of both regularly and geometrically spaced grid lines. The initial zone width for geometric spacing is given by

$$\Delta = \frac{D(R - 1)}{(R^n - 1)} ,$$

where D is the distance being zoned, R is the ratio by which zone sizes change, and n is the number of cells in this segment. It is recommended that values of R close to 1 be used.

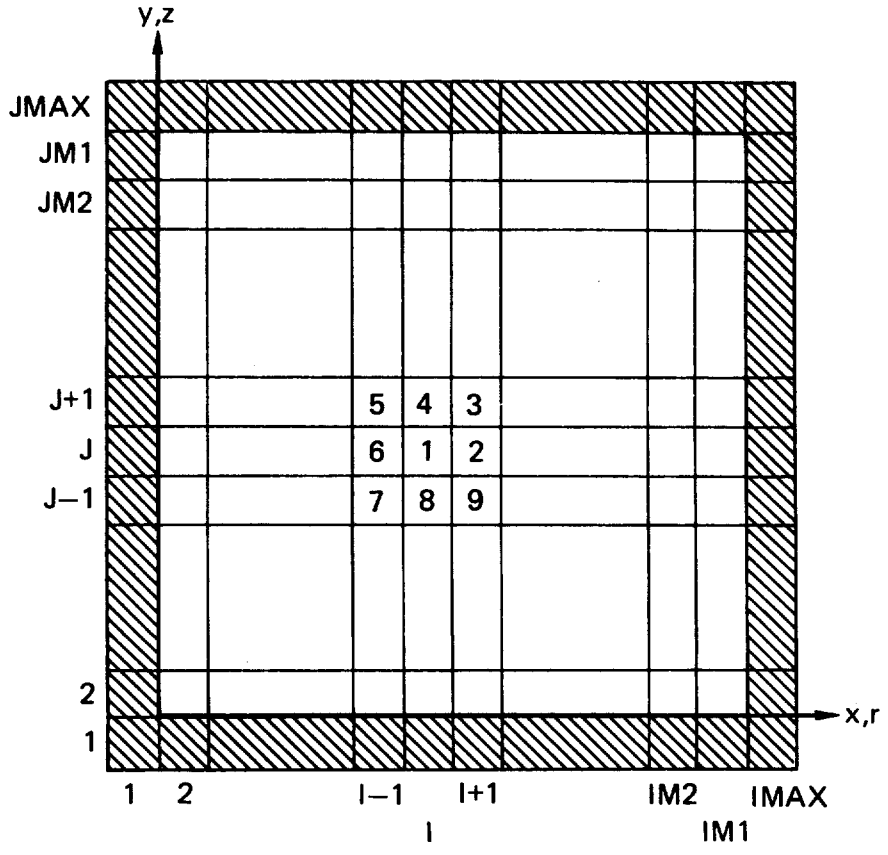


FIG. 8. General mesh arrangement showing the indexing conventions and fictitious boundary cells.

This mesh is laid out as shown in Fig. 8 with the indices and labels used by the code as shown. The problem region is surrounded by a single layer of boundary cells used to impose the desired external boundary conditions.

The numbering of the grid on all input and printouts is that of the user's grid and not the same as used internally by the code.

The indexing for cell (I,J) and its eight nearest neighbors is shown in Fig. 8. This cell is identified by the intersection of I and J grid lines at the upper right hand corner of the cell. The cell center coordinates are designated by XX(I) and YY(I), while the grid line coordinates are given by XT(I) and YT(I). For problems with cylindrical symmetry (axi-symmetric) the X axis becomes the radial coordinate and the Y axis is the axial coordinate.

The components of the fluid velocity, U and V, are specified on cell sides and represent average values of the velocity components normal to their respective side. This placement of the velocity components makes it easy to preserve the conservative property of the differential equations in the finite difference approximation. The cell averaged pressure, P, is associated with the cell center. These variables are shown in Fig. 9 with the notation used by the code.

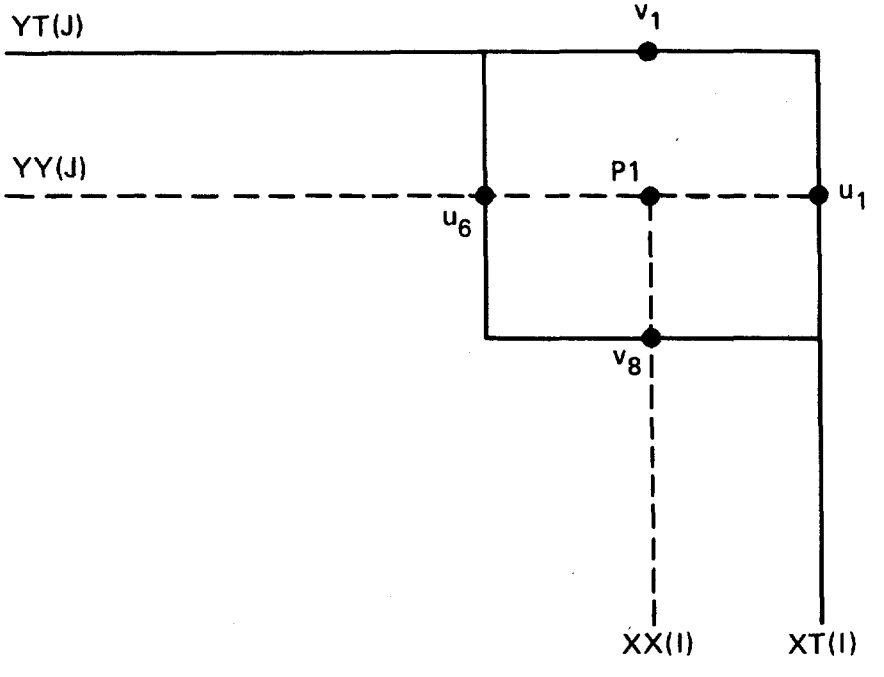


FIG. 9. Location of finite difference variables in a typical cell.

The grid size is indicated by the maximum values, IMAX and JMAX, for the I and J indices, respectively. The numbering system used by the code is sequential by column and thus the index for the cells shown in Fig. 7 are given by

```
K1 = J + (I - 1)*JMAX
K2 = K1 + JMAX
K3 = K2 + 1
K4 = K1 + 1
K5 = K6 + 1
K6 = K1 - JMAX
K7 = K6 - 1
K8 = K1 - 1
K9 = K2 - 1
```

The variables for each cell are stored using the K index. These variables are defined with the notation

P(K)	pressure
U(K)	X-velocity on right side
V(K)	Y-velocity on top side
CI(K)	cell identifier

For a given cell K value, the corresponding grid lines can readily be determined from

```
I = K/JMAX + 1
J = K - (I - 1)*JMAX
```

where integer arithmetic is being used.

3.2 FLOW DIAGRAM

The computer code PELE-IC is compartmentalized into a series of special purpose subroutines (Figs. 10 and 11). This makes it easy to make changes or add new features, since only a few subroutines will be affected. In this section we show how these subroutines are put together to form the code. In the next section the function of each of the major subroutines is briefly described.

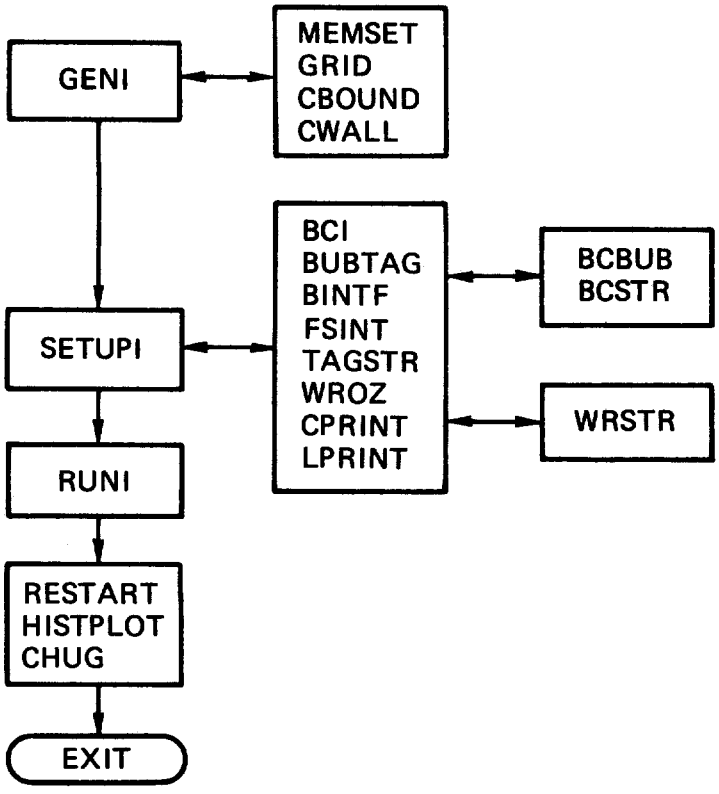


FIG. 10. The main sequence for running from the generator.

3.3 FUNCTIONS OF THE MAJOR SUBROUTINES

A list of each of the major subroutines and a brief description of their functions is given in this section. All subroutines have been given names which indicate their general function.

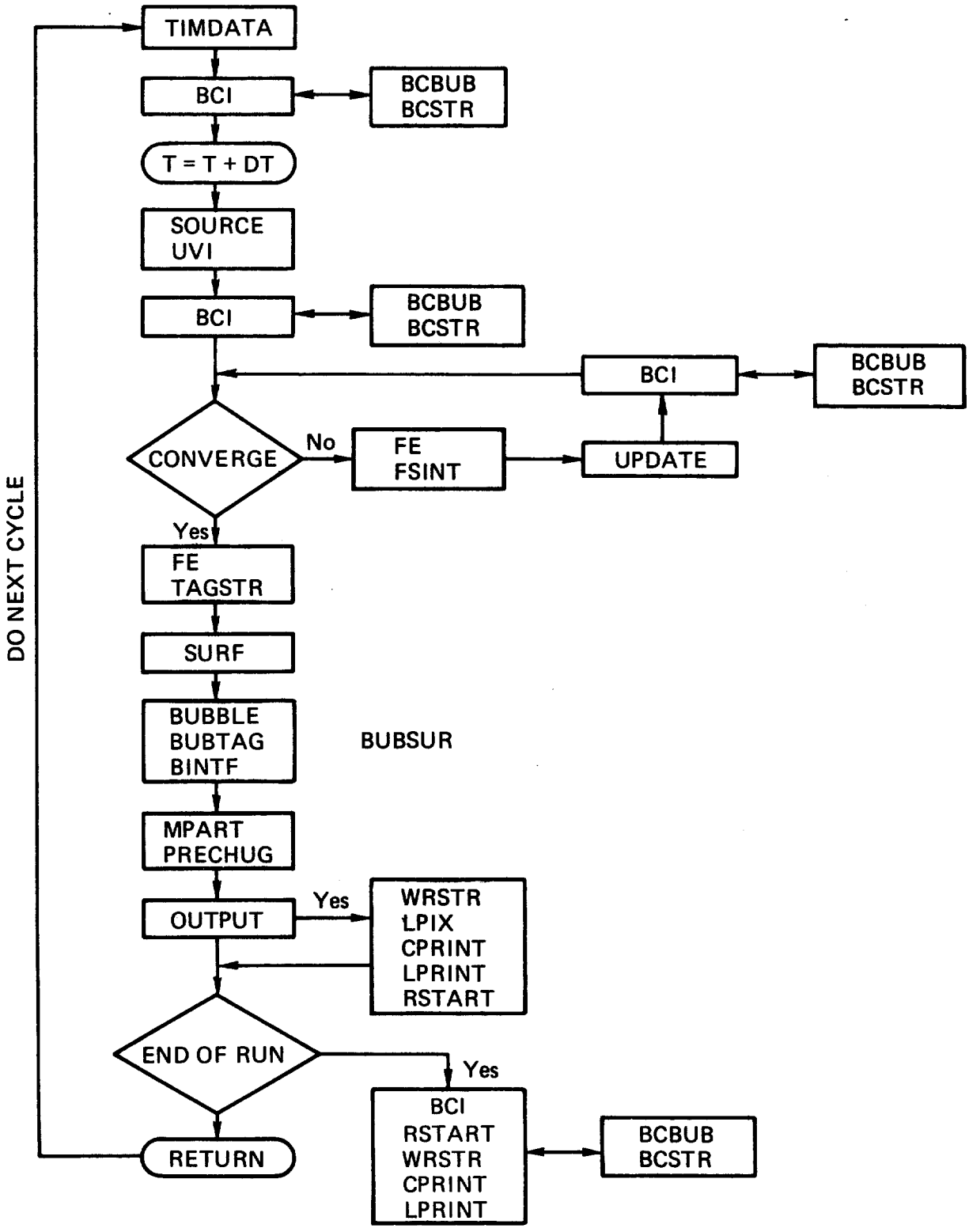


FIG. 11. Processing flow diagram in subroutine RUNI.

BCBUB--BUBBLE BOUNDARY CONDITIONS

The velocity boundary conditions around a free surface are set here. These include the void side velocity and any velocities that are being slaved to neighboring cells.

BCI--SETS UP BOUNDARY CONDITIONS

This subroutine supplies the boundary conditions for the edges of the computational grid, all free surfaces, and rigid structures which follow grid lines. The boundary conditions for arbitrary curved structures in the grid are obtained by a call to subroutine BSCTR. A call to subroutine BCBUB applies the bubble boundary conditions.

BCSTR--CURVED STRUCTURE BOUNDARY CONDITIONS

This subroutine uses the intercept array, BSTR, to set the cell side velocity such that a linear interpolation to the fluid-structure interface matches the normal velocity of the structure. The I or J line velocity is used depending on the slope of the structure element. If uncoupled sides are outside the structure, then their velocities are slaved to the adjacent fluid cell inside the structure.

BINTF--LOCATES THE BUBBLE INTERFACE

Here we find the interface orientation for free surfaces by interrogating neighboring cells. The cell intercept tags MA5-MA8 (see Fig. 13) in variable CI(K) are set to indicate the intersection of the free surface with the cell sides. These intercepts are used by DCTRAN to calculate the advection of fluid.

BSURFP--BUBBLE SURFACE PRESSURE

This subroutine does the interpolation to find the cell centered pressure which gives the prescribed pressure time history on the bubble-fluid interface. The subroutine uses the interface orientation to interpolate from a full fluid cell.

BUBBLE--DOES ADVECTIONS FOR MIXED CELLS

This is the controller for the advection of liquid fractions in mixed cells. It calls subroutine DCTRAN when there is advection between mixed cells. Liquid fractions are advected in order to maintain conservation of mass. After the advection is calculated for all cells, the new void fractions are computed.

BUBSUR--BUBBLE SURFACE

Checks all bubble surface cells for consistency of orientation and tagging.

BUBTAG--SET BOUNDARY TAGS FOR MIXED CELLS

Sets the cell side tags MA1-MA4 for mixed cells. This subroutine is called after we have moved all surfaces.

CBOUND--GENERATE CURVED BOUNDARIES

Generates a circular interface between the liquid and void. The liquid can be located either inside or outside of this circular boundary.

CHUG--CHUGGING EQUATION SOLVER

Main routine that sets up the call to the GEAR package (a differential equation solver) to solve the mass conservation and energy equations for the MIT chugging model of Kowalchuk and Sonin.[3]

CONE--FINITE ELEMENT STIFFNESS MATRIX

Forms the local stiffness matrix and uses subroutine ROTATE and ADD to change to global coordinates and assemble the global matrix, respectively.

CPRINT--LOGIC GRID PRINT OUT OF CELL IDENTIFIERS

Gives a logic grid print out of the ten tags contained in structured word CI(K). This allows the user to know exactly how the code is treating each cell.

CWALL-GENERATES A CURVED WALL

This is the subroutine in the generator package that is used to generate circular or arbitrarily shaped structural boundaries in the grid.

DCTRAN--ADVECTION BETWEEN MIXED CELLS

Does the special fluid advection between mixed cells based on the interface orientation of the donor cell. This algorithm allows us to track the interface without the use of marker particles.

DIFFUN--CHUGGING DIFFERENTIAL EQUATIONS DEFINITION

Provides the mass conservation and energy equation to be solved by the GEAR package.

ERRS--ERROR EXIT

Prints out an error message for both the processor and the generator telling the user where the error occurred.

FE--CONTROLLER FOR COUPLING TO FINITE ELEMENT CODE

Sets up the current pressure loading on the structure elements for use by the finite element code each iteration. Resets the curved structure array, CSTR, after each iteration for use by the fluids code. After convergence, it updates the structures velocity and displacement arrays for the new time step.

FSINT--FLUID-STRUCTURE INTERFACE

This subroutine loads the intercept array, BSTR, from the curved structure array, CSTR, and sets the tag, ICSTR, which indicates the number of cell intercepts on the structure. Whenever two intercepts occur in a single cell, the routine selects the master side using the slope of the element to decide between the I and J intercepts. This master side is then used by the coupling algorithm.

GENI--GENERATOR

Reads and processes all input data for the generation of the problem. The input is handled by a set of thirteen control cards for specifying different types of input information. Calls MEMSET to determine the computer storage requirements for the input grid size.

GMPART--SETS UP MARKER PARTICLES

Sets up different color marker particles in different regions of the grid to monitor the flow of the fluid. This is done as part of the problem generation.

GRID--GRID CREATION

Uses the input to create the grid for the problem. Both uniform or geometric spacing can be specified between grid lines.

HISTPLOT--MAKES TIME HISTORY PLOTS

Makes time history plots of source pressure, top surface level, bottom surface level and pressure for the specified cell.

LPIX--LOGIC GRID PRINT OUT OF CELL OCCUPATION

Gives a logic grid print out showing the relative void fractions of each cell, thus giving a quick reference to the numerical output.

MEMSET--SET MEMORY SIZE

Dynamically sets the computer memory for the size of the problem after the grid size cards are read in.

MPART--UPDATES MARKER PARTICLE POSITIONS

Calculates and relocates the marker particles based on the fluid velocities. The particle velocities used to relocate the particle are computed from an area-weighting scheme involving the nearest cell velocities. The algorithm used is based on that development for the MAC code.

PDATE--PELE DATE

Maintains the creation date of the controllee. It also calls the system subroutine CLOCK to determine the machine and time the run was initiated. This dated version makes it possible to ascertain what changes have been made to the code between problems.

PRECHUG--PELE-IC CHUGGING INTERFACE

This subroutine acts as an interface between PELE-IC and the MIT chugging model. It computes and provides the chugging model with the variables it needs such as total air volume, surface area, average velocity, etc.

RSTART--WRITES THE RESTART DUMP

The restart dumps are identified by the letters PIC and the cycle number. For example; the file PIC100 would contain all the necessary data to

restart the run from cycle 100. These files are also used by the post-processor to obtain graphics.

RUNI--CONTROLLER WHICH RUNS THE CALCULATION

The controller for processing the calculation calls all major computational subroutines and controls print outs and writing of the restart dumps. If a confined ullage is in the problem, its pressure is computed in this subroutine.

SETUPI--SETS UP PROBLEM AFTER GENERATION

Locates and sets indices for the top and bottom surfaces. If the problem has a confined ullage, the initial value is calculated. The hydrostatic pressure is calculated when specified. Calls to subroutines BUBTAG and BINTF set the boundary tags for mixed cells. All structural intercepts are set by calls to subroutines FSINT and TAGSTR.

If a structure has been given an initial loading, then a call is made to subroutine FE to find the static structural solution.

SHELL--CONTROLLER FOR THE FINITE ELEMENT CODE

This is the controller for the thin-shell finite element code. It calls subroutine CONE to form the stiffness matrix, does the Newmark time integration, and calls subroutine SYMBC to invert the stiffness matrix. A switch allows the stiffness reformulation to be bypassed during the fluid-structure compatibility iterations.

SOURCE--APPLIES BOUNDARY PRESSURE SOURCES

Determines the current boundary pressure from the inputed pressure time history. Calls subroutine BSURFP whenever a mixed cell is found.

SURF--COMPUTES NEW SURFACE POSITIONS

Tracks free top and bottom surfaces by explicit liquid advection using the updated velocity field. This is a simpler approach than that used in the BUBBLE subroutines and is applicable only to well behaved surfaces.

SYMBC--FINITE ELEMENT SOLVER

Solves the symmetric banded equations in a single-subscript arithmetic using Gauss elimination.

TAGSTR--SETS TAGS FOR CELLS OUTSIDE STRUCTURE

Sets the tag MA9 = 3 for all cells outside the structure. Other subroutines then skip over all cells with this tag set, thus reducing the number of cells swept during the calculation.

TIMDATA--ACCUMULATE TIME HISTORY DATA

Accumulates time history data for source pressure, top surface level, bottom surface level, and pressure at the specified cell. These data are then used by subroutine HISTPLOT for graphics output.

UPDATE--UPDATES CELL PRESSURE AND VELOCITIES

Updates the pressure field and finds the new velocities during the iteration. Applies the special boundary conditions for free surfaces. Calls subroutine BCI to apply all other boundary conditions and then makes a divergence check on all cells. If the divergence criteria are met, then a flag is set to end the iteration.

UVI--SOLVES THE NAVIER-STOKES EQUATIONS

Solves the Navier-Stokes equations explicitly to obtain the trial velocities, $\tilde{U}$ and $\tilde{V}$, to start the iteration. The differencing uses a variable alpha which weighs the amount of upwind differencing to be used.

WROZ--WRITE OUT "Z" ARRAYS

Writes out the two control arrays, IZ and ZZ, along with the edit selector array, IEDIT.

WRSTR--WRITE STRUCTURE INFORMATION

Calculates the structural stresses and writes out the parameters for the structure's configuration. It also outputs the fluid-structure interface information.

3.4 CONTENTS OF THE DATA ARRAYS

The Intercept Array BCSTR

This array is set in subroutine FSINT and gives the cell-structure intercept values.

BSCTR(L,N)

L The table entry and has values 1-ICSTR

N = 1 Structured word SWB

2 F = intercept value measured in the positive direction

3 Tangent of the slope angle of the element

4 US = structure X-velocity at intercept

5 VS = structure Y-velocity at intercept

SWB structured word with components

MB1 0 = I intercept, 2 = J intercept
MB3 K = cell number
MB4 structure element number

The interpretation of these values is shown in Fig. 12.

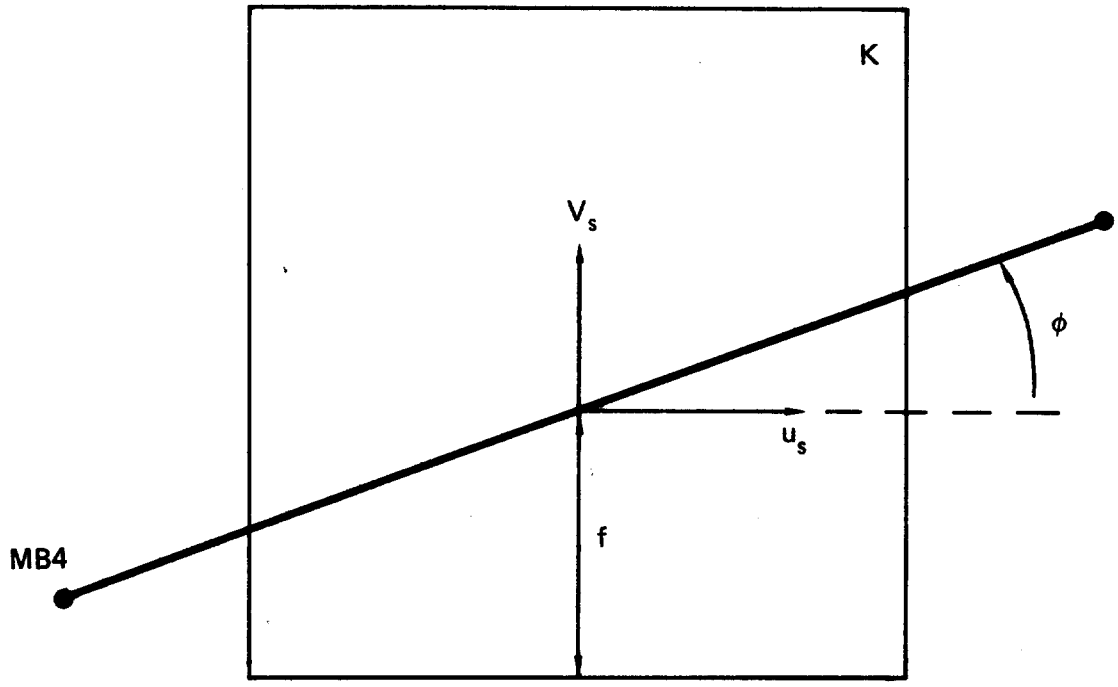


FIG. 12. Values of the cell-structure intercept values stored in array BCSTR.

Structure Nodal Arrays, CSTR, RSH, and ZSH

These arrays are created during generation of the problem. The CSTR array is updated by subroutine FE to give the current values associated with the structural nodes.

RSH(NP) original X coordinate of the nodes
ZSH(NP) original Y coordinate of the nodes
CSTR(N,L) current values

L The table entry and has values 1-NP

N = 1 X coordinate

2 Y coordinate

3 US = X velocity component at node

4 VS = Y velocity component at node

The current values can be given an initial displacement during generation of the problem.

Surface Arrays HT and JT, HB and JB

HT(I3MAX) Top Surface Array

This array is divided into three sections each IMAX long and each value corresponds to a given I value.

1. Current location of top surface.
2. Previous location of top surface last cycle.
3. Pressure applied to the surface.

JT(I2MAX) Top Surface Index

This array is divided into two sections and identifies the cell with the top surface.

1. The cell J value.
2. TAG 0 = use subroutine SURF to move surface
   1 = use subroutine BUBBLE

The second array of tags is used in vent clearing problems. While the water level is initially in the pipe, it is treated as a simple surface. Once vent clearing occurs, this tag is set and the bubble algorithm controls the surface motion.

A surface is considered to be a cell J if

$YT(J - 1) < HT \leq YT(J)$.

HB(I2MAX) Bottom Surface Array

This array is divided into two sections each IMAX long and each value corresponds to a given I value.

1. Current location of bottom surface.
2. Previous location of bottom surface last cycle.

JB(IMAX) Bottom Surface Index

This array identifies the cell with the bottom surface by specifying its current J value.

A surface is considered to be in cell J if

$YT(J - 1) \leq HB < YT(J)$.

Cell Identifier Array CI

In addition to the primitive variables, each cell has associated with it a structured word containing ten tags used to identify the status of the cell and each of its four sides. The cell sides are numbered counterclockwise starting on the right as shown in Fig. 13.

The ten tags are designated as follows:

MA1-MA4 Cell Side Tags

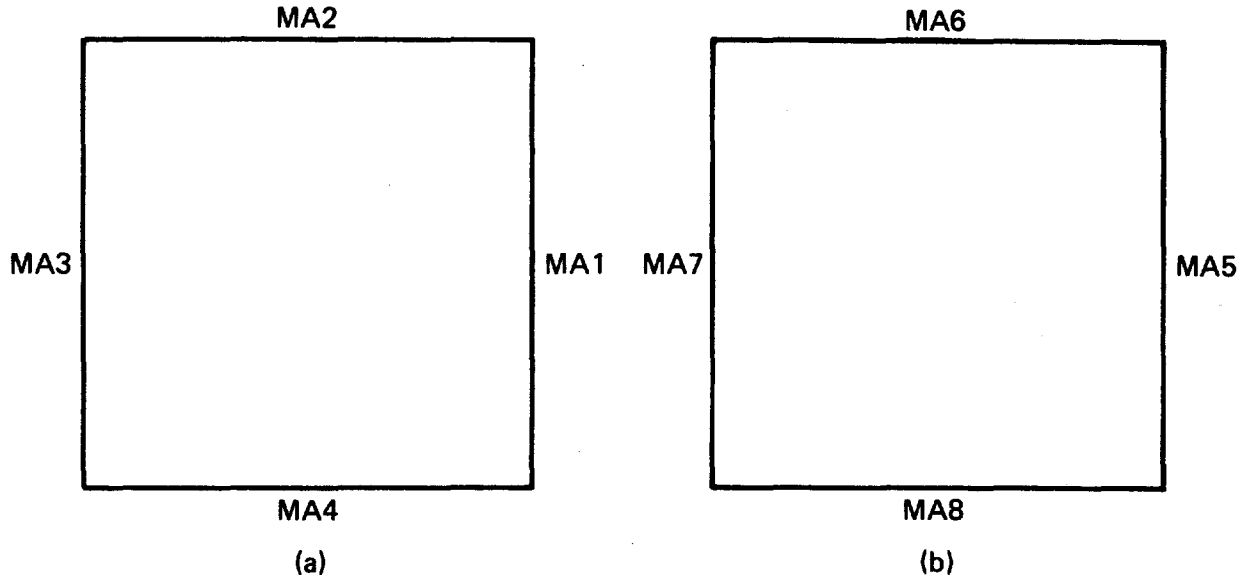


FIG. 13. Location of tags for cell sides: (a) border tags and (b) intercept tags.

These tags indicate the status of the cell sides and the values assigned have the following meaning:

1 = rigid boundary
2 = liquid side
3 = mixed side
4 = void side.

The values 2-4 are used only for mixed cells. The rigid boundary designation is used for all cells and is used to require that the velocity component on this side is set to zero.

MA5-MA8 Fluid Intercept Tags

These tags correspond to sides one to four in a counterclockwise rotation. They give the positive fluid-void fractional intercepts for each side designated as a mixed side by the cell side tags. They are given values from 1-10 and serve as pseudo marker particles in the determination of the

location of free surfaces. For example, an intercept value of 6 means that the interface is located between 0.5 and 0.6 of the side length measured in the positive direction.

MA9 Location Tag

This tag tells the code whether the cell is inside or outside a structure. It can have the two values

0 = inside
3 = outside.

The calculation sweep ignores all cells tagged as outside.

MA10 Structure Intercept Tag

When a structural element lies within a cell, this tag gives the number of I and J intercepts and thus can have values from 0 to 2.

4.0 PROBLEM GENERATION

4.1 INPUT CONTROL CARDS

The code generation of a PELE-IC problem is controlled by a set of 13 control cards, which have the functions as outlined below:

-1 CONTROL TO READ IN THE IZ CONTROL ARRAY

+0 END OF THE INPUT DECK

+1 CONTROL TO READ IN THE ZZ CONTROL ARRAY

+2 CONTROL TO READ IN TOP SURFACE.USED WHEN THE TOP SURFACE OF THE INCOMPRESSIBLE FLUID IS NOT AT THE TOP OF THE GRID

+3 CONTROL TO READ IN BOTTOM SURFACE.USED WHEN THE BOTTOM SURFACE OF THE INCOMPRESSIBLE FLUID IS NOT AT THE BOTTOM OF THE GRID

+4 CONTROL TO READ IN BOUNDARY SOURCES.USED FOR THE VARIABLE PRESSURE DRIVING FORCE

+5 CONTROL TO READ IN INTERIOR BOUNDARIES ON GRID LINES.ALSO USED TO OR TO SPECIFY THE GRID LINE USED TO DESIGNATE A PIPE

+6 CONTROL TO SET INITIAL CONDITIONS

+7 SPECIFY CELLS TO BE MONITORED EACH CYCLE

+8 STRUCTURES COUPLING

+9 CONFINED ULLAGE

+10 MARKER PARTICLES

+11 ARBITRARILY SHAPED BOUNDARY

+12 EDIT SUPPRESSION

4.2 PROBLEM SETUP FOR PEPE-IC

Below we give a detailed outline of the various input parameters used in the generation of a problem to be run on the PEPE-IC code. The code accepts free form input; therefore, it is not necessary that the numbers be in any specific column.

PROB(3) Problem name

IMAX, JMAX
'X' DIST NZONE R
'Y' DIST NZONE R
0 END GRID INPUT

PROB(3)	Name can consist of three words with a maximum of 8 characters each.
'X'	indicates zoning in X-direction
'Y'	indicates zoning in Y-direction
DIST	final coordinate for a given interval. The interval will start at 0, or the previous DIST value
NZONE	number of zones in the interval
R	0 or 1 indicates equal zoning .GT. 0 geometric zoning increasing with ratio R .LT. 0 geometric zoning decreasing with ratio R

NOTE: The total number of zones must equal IMAX and JMAX for the X and Y directions.

-1 CONTROL TO READ IN THE IZ CONTROL ARRAY

IA,IB,IV(10)	DATA CARD	
	IA	data entered starting at IZ(IA)
	IB	no. of data values in IV(10)
	IV(10)	values to be entered
	IV = 0	exit this control

The values of the "IZ" control array are listed below along with their default values. Not all values are entered by the user with this control.

Some are set by other control inputs. The controls not initially set using this input option are marked with an asterisk (*). This list gives the user the interpretation of the printout.

	IZ	SYMBOL	DEFAULT	USE
*	1	IMAX	0	zones in X-direction
*	2	JMAX	0	zones in Y-direction
	3	IBL	1	left boundary tag
	4	IBR	1	right boundary tag
	5	IBT	1	top boundary tag
	6	IBB	1	bottom boundary tag
*	7	IST	0	top surface tag
*	8	ISB	0	bottom surface tag
	9	IGPO	0	1 = give cycle zero printout
*	11	IPP	0	pipe tag
*	12	IOP	0	initial I line of pipe
*	13	ILP	0	last I line of pipe
*	14	JOP	0	initial J line of pipe
*	15	JLP	0	last J line of pipe
	16	IHT	0	tag for time history plots of HT(IHT)
*	17	ISTR	0	tag for flexible structure
	18	IPLOT	0	I index for time history pressure plots
	19	JPLOT	0	J index for time history pressure plots
	20	KREST	1	tag for restart dumps
*	21	IULL	0	tag for confined ullage--set by control 9
	22	NXB	0	marker particles per cell, X-direction
	23	NYB	0	marker particles per cell, Y-direction
	24	NPP	0	total number of marker particles
	25	ICB	0	flexible bottom plate I value
	26	ITIM	0	0 = cycle check, .NE. 0 = time check
	27	IPL2	0	same as IPLOT
	28	JPL2	0	same as JPLOT

	29	IPL3	0	same as IPLOT
	30	JPL3	0	same as JPLOT
	31	IPL4	0	same as IPLOT
	32	JPL4	0	same as JPLOT
	33	IFE1	0	time plot of finite element node displacement
*	34	ICSTR	0	no. of grid intercepts with structure
*	35	NP	0	no. of finite element nodes
	36	ICHUG	0	tag for the chugging routine
*	38	ISWP	0	single value structure tag
*	39	IFL	0	tag for fluid structure orientation
*	40	MCELL	0	number of monitored cells
	41	IOFSB	1	initial I for free surface on bottom
	42	ILFSB	IMAX	last I for free surface on bottom
	43	TTY	0	tag for TTY cycle monitoring
	44	IFE2	0	time plot of finite element node displacement

The following values are entered by the user with this control unless the default values are desired.

3-6 BOUNDARY TAGS IBL, IBR, IBT, IBB

1 rigid--free slip
2 rigid--no slip
3 continuative (flow thru)
4 periodic.

7-8 SURFACE TAGS IST, ISB

0 no surface or has a curved structure
1 free surface
2 rigid boundary--uses Newtron-Raphson iteration.

9 CYCLE ZERO PRINT OUT TAG, IGPO
0 will suppress a cycle zero printout
1 gives a cycle zero printout.

16 HT is the array containing the top surface.
IHT is the I value of the surface column to be monitored.

20 RESTART TAG KREST
0 skip restart dump
1 make restart dump and print out
2 make restart dump only

33,34 IFE1 AND IFE2 give the finite element nodes for time history plot.

43 TTY TELETYPE CYCLE MONITORING

+N print out to TTY every N cycles
-N print out to TTY every N cycles after use of SW2

Error exist 1 and 2

+1 CONTROL TO READ IN ZZ CONTROL ARRAY

IA,IB,DUM(6) DATA CARD
SAME AS CONTROL -1

The values of the "ZZ" control array to be set or changed by the user are listed below along with their default values.

ZZ	SYMBOL	DEFAULT	USE
3	DELT	0.0	Time step, δt
4	XNU	0.0	Kinematic viscosity, ν
5	CYL	0.0	Symmetry tag 0 = plane, 1 = cylindrical
6	EPSI	1.0E-4	Convergence test for divergence
7	SPEED	0.0	Sound speed in water (148 cm/msec)
8	DTMAX	0.0	(Set automatically)
9	GY	0.0	Gravitational constant (negative)
10	OMG	1.3	Over-relaxation factor (pressure change)
11	ALPHA	0.2	Donor cell differencing
15	GAMMA	1.0	Over relaxation factor (Newton-Raphson iteration)
16	CFIN	1.0	End cycle
17	CPRT	0.0	Cycle for first print out
18	DCPR	0.0	Print out cycle frequency
19	VAPP	0.0	Water vapor pressure
20	BUBP	0.0	Initial bubble pressure
21	BUBG	0.0	Gamma for bubble gas
22	BUBV	0.0	Original bubble volume (done by code)
23	PIPE	0.0	Bottom of pipe
25	PLATE	0.0	Coordinate of bottom plate
28	UVOL	0.0	Total confined ullage volume
29	UPRO	0.0	Initial absolute pressure of ullage
30	UGAM	0.0	Ratio of specific heats of ullage gas
31	UVOLO	0.0	Ullage volume outside grid /2π (CYL = 1.) or /1(CYL = 0.)
33	TMPIN	0.0	time for marker particle input
34	DTMP	0.0	Delta-t for marker particle input
38	CRAD	0.0	Radius of cylindrical wall

Guidelines and Comments

ALPHA In general .GT. (MAX VEL)*$\delta t/\delta z$

0 = space centered differencing

1 = full donor cell differencing

An input of negative ALPHA will cause the code to use ABSF(ALPHA) for the first cycle and then compute a variable ALPHA for each subsequent cycle depending upon the maximum velocity in the grid.

$$ALPHA = 1.3*MAX(|U*DT/DX|, |V*DT/DY|)$$

with the maximum value allowed being ALPHA = 1.

NOTE: Space centered differencing of advective terms is generally unstable. Partial donor cell differencing results in less numerical diffusion than full donor cell differencing.

DELT .LT. 0.4*MIN $[\delta X/|U|, \delta y/|V|]$

NOTE: The time step is limited by the flow field so that the fluid can transverse less than half a cell during each time step. Even smaller time steps may often reduce the total number of iterations required for a problem.

EPSI Always should be $\leq$ 1.0E-3

NOTE: Mass conservation is affected by the choice of EPSI. The maximum mass error is given by

(EPSI)*(No. of cells)*(No. of time steps).

The maximum error will occur only if the sign of the divergence in each cell is of the same sign each cycle.

Since the value of EPSI is used in the divergence test during the iteration to determine the pressure field, it must be consistent with the velocity flow field. This is of special importance when coupling to very slowly moving structures.

GY — If a +1 is entered then the default value of -9.81E-4 for the gravitational constant will be entered.

OMG — Never >2

XNU — $< \frac{0.5}{\delta t} \left[\frac{\delta x^2 \, \delta y^2}{\delta x^2 + \delta y^2} \right]$

Error exit 3

+2 — CONTROL TO READ IN TOP SURFACE

IST — type of surface

IST

- 0 no surface
- 1 free surface
- 2 rigid boundary

IA,IB,DUM(5) — data card

IA — entry control

- -1 all cells at same level DUM(1)
- +1 enter level for each cell
- 0 exit this control

IB — number of data values in DUM(5) to be loaded starting at cell IA

DUM(5) — Surface level for each cell. The level does not have to be on a grid line.

Error exit 4

+3	CONTROL TO READ IN BOTTOM SURFACE

ISB,IO,IL	type of surface
IA,IB,DUM(5)	data card

Same as control +2

ISB

- 0 no surface or a flexible structure
- 1 free boundary
- 2 rigid boundary

IO, IL I values for free surface
IF = 0 then default will be set to
IO = 1 and IL = IMAX

Error exit 5

+4	CONTROL TO READ IN BOUNDARY SOURCES

SORCE(9)	source control numbers
T, P	source table (omit if ISO = 3)
	Last card must have T = 0 to signal end of input
	a maximum of 45 (T,P) values can be given.

SORCE

1 ISO Type of source

- 0 no source
- 1 source is constant pressure in time
- 2 source is variable pressure in time
- 3 bubble pressure + water vapor pressure
- 4 ullage source

		5 U-velocity on grid line
		6 V-velocity on grid line
		7 surface cell velocity
2	IO	I sweep--first index
3	IL	I sweep--last index
4	JO	J sweep--first index
5	JL	J sweep--last index
6	IIO	I sweep--first index
7	IIL	I sweep--last index
8	JJO	J sweep--first index
9	JJL	J sweep--last index

IO, IL, JO, JL define the grid area containing the source. If a second pipe is to be specified, then use the set SORCE(6)-SORCE(9) to specify its range.

For pipe venting problems: IO, IL specify the range of the pipe and then input JO = JL = 0. This will cause the source to be applied internally to a growing bubble which was created by a vent clearing.

This allows the code to use separate boundary conditions for the top surface and the bubble.

If the problem contains a top surface and a bubble which was not created by vent clearing, then set IO = IL = JO = JL = 0. Again this allows the use of two sets of boundary conditions.

ISO = 3, BUBBLE PRESSURE

This option gives a variable bubble pressure depending on volume changes. Under control +1 you must enter

ZZ(19) VAPP
ZZ(20) BUBP
ZZ(21) BUBG

The code will then calculate the original bubble volume and store it as

ZZ(22) BUBV

ISO = 4 ULLAGE SOURCE

This option allows you to prescribe a variable ullage pressure.

ISO = 5,6 GRID LINE VELOCITIES

These options specify velocities at the boundary of the grid.

ISO = 7 SURFACE VELOCITY

This option will prescribe the velocity V8 for a cell with a top surface.

+5	CONTROL TO READ IN INTERIOR BOUNDARIES ON GRID LINES OR TO SPECIFY THE GRID LINE USED TO DESIGNATE PIPE

I0,IL,J0,JL,IP	Range of boundary
MA1,MA2,MA3,MA4	Interior boundary conditions (Not used if IP = +1 or -1)

These two cards must be used for each grid line specified. Note that a grid line borders two cells.
I0 = 0 Exit this control

Pipe tag IP

0	Boundaries do not represent a pipe
-1	Boundary represents left side of pipe
+1	Boundary represents right side of pipe

NOTE: A pipe is defined as having both sides in the fluid, i.e., a double sided boundary involving two adjacent cells. If the pipe is to be evacuated and a bubble blown, then this control must be used.

When IP = -1 or +1 then the code automatically sets the adjacent cell.

If the pipe is on axis, only use the +1 entry. A one sided pipe, i.e., one with one side outside the fluid can be defined using control +11.

Boundary Conditions, MA1-MA4

0 no boundary
1 rigid--free slip

Once a grid line has been designated as rigid, it remains so throughout the calculation.

Error exit 6

+6	CONTROL TO SET INITIAL CONDITIONS
IO,IL,JO,JL	Range for initial conditions IO = 0 end this control
IA,VAR	Input variable for this range IA tag identifying variable, VAR 0 end this range 1 pressure 2 U-velocity 3 V-velocity

4 void fraction, VF

0 = liquid, 1 = void

5 cell is tagged as outside; i.e., MA9 = 3

Outside structure on liquid side must be VF = 0.

+7 SPECIFY CELLS TO BE MONITORED EACH CYCLE

I,J Cell coordinates

I = 0 indicates end of entries

If I is negative then we monitor the structure element

L = -I

Up to 15 cells can be monitored

This option provides a cycle by cycle print out of all variables for the cell or element selected for monitoring.

Error exit 8

+8 STRUCTURES COUPLING

TH, GNU, EE, RO, DEL, IB, NP, ICP

IN1, ICFE(1-6)

IN2, SX, SY, ST (For element IN2)

IN3, TH(IN3)

This entry automatically sets ISTR = IB + 1 and ISB = 0

Th plate thickness

GNU Poisson ratio

EE Young's modulus

RO plate density

DEL structure damping factor

IB tag 0 = axisymmetric, 1 = planar, 2 = beam

NP number of finite element nodes

ICP type tag

TH +value will set all elements to same thickness
-value requires thickness array for all elements

DEL gives % of critical damping
typical values range from 0.01 to 0.1

NP If NP = 0 then code sets NP equal X grid size

ICP -1 cylindrical structure at the cell center of cell I
0 general structure--must also use control 11
+J flat plate at bottom of cell J
sets ISWP = -1

. . . . . .

IN1 +1 = enter tags and then read next card
0 = exit
-1 = read next card

A value of 0 or -1 without the use of a +1 will use the default values of ICFE.

ICFE end tags for shell default values = 0,0,0,1,1,1

The first three values are for the left (or bottom) end, and the final three values are for the right (or top) end of the structure and refer to the three degrees of freedom: X, Y, and angle.

0 = free, 1 = fixed

. . . . . .

IN2 +L = element number
0 = exit
-1 = read next card
SX stiffness per unit radian in X direction
SY stiffness per unit radian in Y direction
ST stiffness per unit radian in theta direction

A value of 1.E20 gives the same result as the tag ICFE = 1.

. . . . . .

IN3 +L = element number
0 = exit
TH thickness of element IN3

+9 CONFINED ULLAGE

UVOLO, UPRO, UGAM

This entry automatically sets IZ(21) = IULL = 1

UVOLO ullage volume outside grid for cylindrical symmetry divided by 2π
UPRO initial ullage absolute pressure
UGAM ratio of specific heats

If not entered then default values are:

UPRO = 1.0
UGAM = 1.4
The ullage pressure is calculated as
UP = UPRO*(UVOL/VOL)**UGAM

Where VOL is the current volume and UVOL is the initial ullage volume.

This option is used when there is a large confined air space in a vessel and it is not desired to have the computational mesh cover space which will never have fluid flow. This allows the code to calculate the pressure change based upon volume changes.

If the problem does not fill up the grid, then use a negative volume for that part outside the problem but inside the grid.

When using a variable ullage pressure, then all pressures in the problem must be absolute.

+10 MARKER PARTICLES

TYPE, NXB, NYB, TPARTIN, DTPART, COLOR

or

TYPE, NX, NY, XC, YC, XD, YD, COLOR

'TYPE' indicates type of marker particles setup

0 exit
1 source input on BDY
2 initial set up at time 0 for region specified.
(XC,YC) and (XD,YD) must be center of cell.

TYPE = 1 NXB = number of particles per cell (X-direction) on BDY or
NYB = number of particles per cell (Y-direction) on BDY
TPARTIN = time for source input of marker particles
DTPART = DELTA-T input of marker particles
(not currently available)

TYPE = 2 NX = number of particles per cell (X-direction)
NY = number of particles per cell (Y-direction)
XC = left-most X-center coordinate of region

YC = low-west Y-center coordinate of region
XD = right-most X-center coordinate of region
YD = highest Y-center coordinate of region
Color = color of marker particles
Color available : yellow, green, cyan, magenta, white

NOTE: Maximum of 15 sets of marker particles allowed, and a total of 1000 particles.

+11 ARBITRARILY SHAPED BOUNDARY

IO, IL, JO, JL, JCB
RAD, RC, ZC, INC, IOUT -- FREE SURFACE JCB = 0
or
IO, IL, JO, JL, JCB
RAD, RC, ZC, NEL, IFL -- STRUCTURE JCB = 1
LAST

IO first index -- I sweep (X)
0 = exit
IL last index -- I sweep
JO first index -- J sweep (Y)
JL last index -- J sweep
IO, IL, JO, JL define region of grid containing surface
JCB Type of surface
0 = free, 1 = rigid or structure
RAD radius of circle
0 = arbitrary surface read in and skip to option below
RC X coordinate of center of circle
ZC Y coordinate of center of circle

INC inside tag

```
IOUT      outside tag
                -1 = void, 0 = liquid
NEL           number of elements--JCB = 1
              not used for RAD = 0
IFL           fluid-structure orientation
                   0  fluid inside
                   1  fluid outside
LAST = 0  exit
     = 1  read in initial displacements
          (see instructions below)
+ + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + +
```

Arbitrary surface, for RAD = 0.

This option allows the user to generate arbitrarily curved boundaries which need not fall on grid lines. The structure may be double valued in X, but must be single valued in Y.

for RAD = 0 then add the following cards as needed

IN, X, Y

IN number of elements from last coordinate
X X coordinate
Y Y coordinate

NOTE: first card must have IN = 1
last card must have IN = 0, -1, -2
-1 sets ISWP = +1
-2 sets ISWP = -1

ISWP single value structure tag

If a structure is anywhere single valued in X, then you must tell the generator which side is outside the fluid.
+1 = top side
-1 = bottom side

```
+ + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + + +
```

Initial displacements when LAST = 1

IN, DX, DY

IN node number

0 = exit

DX X displacement

DY Y displacement

+12 EDIT SUPPRESSION

I1 I2 Etc. 0

Entry of a number will suppress print out.

1 logic picture

2 cell identifiers

3 void fractions

4 pressure

5 U-velocity

6 V-velocity

7 divergence

4.3 SAMPLE INPUT DECKS

In this section we give several sample input decks to guide the user in generating a problem.

Funnel

This problem is that of a draining funnel. It exercises the arbitrarily shaped boundary option of the code.

```
FUNNEL 10/19
     10   30
   X    10.          10.     0.
   Y    30.          30.     0.
   0
-1
   6    1    3
   9    1    1
   0
+1
   3    1    .3
   6    1    1.E-7
   9    1    -9.81E-4
   5    1    1.
   16   3    5.     1.     1.
   0
+2
   1
  -1    0    20.
   0
+3
   1    1    1
  -1    0    0.0
   0
+4
   1    1    10   11     20
   5000.     1.
   0
+5
   1    1    1    10     1
   0
```

```
+6
   1    1    1    10
   4    0.
   0
   0
+11
   1    10   10   22    1
   0.
   1    1.   10.
   3    10.  20.
   1    10.  22.
   4    0.   22.
  -1
   0
+0
```

Growing Spherical Bubble

This problem is that of growing bubble with a constant driving pressure. It exercises the free surface algorithm of the code.

```
GROWING BUBBLE 12/19/78

     20   40
   X    20.0  20.   0.
   Y    40.0  40.   0.
   0
-1
   4    3    3     3     3
   16   1    2
   20   1    1
   43   1    1
   0
```

```
+1
   3    1    0.08
   5    1    1
   16   3    100.  25.   25.
   0
+4
   2    1    20    1     40
   1       0.2
   1000.   0.2
   0.0
+11
   1    3    18    26    0
   2.0       0.          20.   -1   0
   0
   0
+0
```

Submerged Vibrating Pipe

This problem is that of a submerged flexible pipe excited by a pressure applied to a circular water surface. It exercises the curved structure interaction algorithm.

```
SUBMERGED PIPE 12/08/78

     10   20
   X      100.  10.   0.
   Y      200.  20.   0.
   0
+1
   3       1     .4
   5       1     0.
   6       1     1.E-6
   16      3     3     1     1
   0
```

```
+4
   2     1     10     1      10
   0.    0.
   500.        0.1
   0
+7
  -1
   0
+8
         .635 0.30   2.0685E6   7.803     0.    0  0  0
   1        1 0 1    1 0 1
   0
+11
   1     10    1      20     0
        82.80 0        100.    0  -1
         1 10 1   20     1
        25.72 0    100.       11    1
         0
+0
```

5.0 RUNNING THE CODE AND POSTPROCESSING

5.1 OBTAINING THE PELE-IC CODE

Library Containing the Controllee

This section describes the access to the PELE-IC code on the Lawrence Livermore Laboratory CDC-7600 computer operating under LTSS.

The code is stored in a "TAKE" directory in a "LIX" file named IPELEXXYY where XX and YY are the month and date of the version it contains. It is obtained by using the ELF routine:

```
User:  ELF
User:  LST    .593800:SOLA:PELE
```

The command gives a listing of the files from which the user can select the latest version using the command

```
User:  RDS    .593800:SOLA:PELE:IPELEXXYY
User:  END
```

The "LIX" file IPELEXXYY is the master library and contains several files. The ones of interest to the user are PELEIC and MAPELEIC which are the controllee and the memory map, respectively. To get these files use the LIX routine

```
User:  LIX IPELEXXYY
User:  GR PELEIC MAPELEIC
       END
```

Other files used for code development are:

SPELE A "LIX" file containing all the FORTRAN listings.

BPELEIC The binary file of the code after compiling on CHAT and assembled using the ALTER routine.

IORANA The cliche used by the fluids code.

FES The cliche used by the finite element code.

Library Containing the Users Guide

The latest users guide is also contained in a "LIX" file named USERXXYY in the same way that the controllee is stored.

It is obtained by using the ELF routine:

```
User:  ELF
User:  LST    .593800:SOLA:MANUAL
```

The command gives a listing of the files from which the user can select the latest version using the command:

```
User:  RDS    593800:SOLA:MANUAL:USERXXYY
User:  END
```

This file contains several sections which can be accessed individually. If the user desires the entire set consolidated then he performs the following operation:

```
User:  EXE APTLIB MLIB USERXXYY
    :  ALLOUT HSP USERXXYY* BOX ZZ ID
```

Additional instructions on time history plotting and Fourier analysis can be obtained from E. Y. Gong, ext. 2-7214

5.2 RUNNING THE PELE-IC CODE

PELE-IC runs under two modes on the CDC-7600 and the Cray computers:

A) from the generator which creates cycle 0
B) from a restart dump.

A PELE-IC run terminates when the specified end cycle has been reached. It can also be terminated by use of sense switch one by entering the command SW1 on the teletype.

At any time the code is running, the user may enter any of the following special commands:

SW1 terminate the problem run

SW2 turn off or turn on the TTY print out as specified by the variable IZ(43) = TTY

SW3 print out the time, cycle, total number of iterations and the iteration count for the current cycle. The code will query the user for "SW1", "DUMP", or "LF".

"SW1" will terminate the run.
"DUMP" will produce a restart dump at the current cycle.
"LF" the line feed command will continue the run.

LF will cause a print out as in SW3 but will automatically continue the problem.

NOTE: The symbols, / t v , refer to the Livermore time sharing system (LTSS) running time and priority value designations.

* *

Method A--Run from a Generator Deck

The control of the processor is set from the values in the input deck. The variable CFIN indicates the final cycle to be calculated followed by the

creation of a restart dump for the given cycle. An input deck must be supplied by user. Default name will be PELED. An input deck other than PELED may be used by typing the name after GEN.

```
User:    PELEIC / t v
TTY:     DUMP-FILE RUN-TO-CYCLE DUMP-CYCLE-INTERVAL OR GEN ??
User:    GEN
TTY:     RESTART DUMP  :  PIC100 WRITTEN AT CYCLE 100 TIME = 8.E2
         ALL DONE
```

At the end of a problem run, a plot file called: UXA80 is produced which contains time history plots of variables specified through input. The routine, UXTV, may then be used to view the plots and produce hard copies. UXTV is a utility routine that can be obtained by typing

```
EXE MJBLIB UTXV DR./ t v
```

For instructions on Cray time history plots, see E. Y. Gong, ext. 2-7214.

Method B--Run from a Restart Deck

Instead of running from cycle 0 with an input deck, the user runs a problem further in time from a given restart dump. Suppose a restart dump PIC100 is produced from method (A). The user wishes to continue running from cycle 100 to 140 with restart dumps produced at every 20 cycles.

```
User:    PELEIC / t v
TTY:     DUMP-FILE RUN-TO-CYCLE DUMP-CYCLE-INTERVAL OR GEN ??

User:    PIC100 140 20
TTY:     RESTART DUMP: PIC120 WRITTEN AT CYCLE 120 TIME = 8.5E2
         RESTART DUMP: PIC140 WRITTEN AT CYCLE 140 TIME = 9.0E2
         PROBLEM IS FINISHED AT TIME -9.0E2 CYCLE = 140
         ALL DONE
```

At the end of a problem run, a plot file called: UXA80 is produced which contains time history plots of variables specified through input. The routine, UXTV, may then be used to view the plots and produce hard copies.

5.3 POSTPROCESSING CAPABILITIES

The postprocessor is a program which will take a restart dump from the PELEIC processor and produce edits and CRT plots and the name of the program is POSTIC, and can be obtained from the photostore directories.

.323591:PELEIC:POST:POSTIC

There are three main branches in POSTIC:

BRANCH-1 : EDIT
BRANCH-2 : PRINT
BRANCH-3 : CRT

The types of services the user may perform under EDIT are:

1) print out of dump to the high speed printers
2) interactively change values in the dump.

The PRINT branch will produce a print out of the dump. CRT is the main graphic section in POSTIC. There are three main CRT options:

1) velocity flow field plots
2) contour plots
3) variable plots.

The following pages will show how the user interacts with the post processor to produce edits and CRT plots. The user must have both the program POSTIC and a restart dump (e.g., PIC100) on disk before she starts executing. To start execution, type the following:

```
User  :  POSTIC PIC100 V84 / t v
TTY   :  TYPE EDIT, PRINT, CRT OR END
```

* *

EDIT OPTION

* *

The edit option allows you to display and/or change values in the restart dump. As you choose the option to be performed, you enter a certain level in the code. To exit from this level, type 'END'.

```
User  :  EDIT
TTY   :  TYPE : NAME INDEX OR 'CHANGE' NAME INDEX NEWVALUE OR END

User  :  IZ 5
TTY   :  1
         ?

User  :  P IJ
TTY   :  5.878806E-2
         ?

User  :  CHANGE IZ 5 7
TTY   :  IZ(5) CHANGE TO 7

User  :  END
TTY   :  PIC100 UPDATED TO : PIC100*
         TYPE EDIT, PRINT, CRT OR END
```

NOTE: The legal variables are NAME, PROB, IZ, IDUM, ZZ, DUM, P, U, V, VF, CI, HB, JB, HT, JT, XX and YY. The '?' indicates you are in the edit mode. Typing an 'END' will direct you out of this mode.

* *

PRINT OPTION

* *

The print option will produce a print out of the entire restart dump. The name of the print file will be HX(---) where (---) is the cycle number from the dump.

User : PRINT
DISK : - FILE : HX100 is produced -
TTY : TYPE EDIT, PRINT, CRT OR END

* *

CRT OPTION (TELEVISION TMDS DISPLAY)

* *

A. General Features

The CRT option allows the user to obtain contour, velocity or variable plots. The user may specify where he wants the output to go. The available devices are:

TMDS-televsion display monitor
RJET-hard copy at a remote job entry terminal
DD80-hard copy from high speed output
UX80-CRT files for UXTV
FR80-105MM microfiche
C16-16mm color film
C35-35mm color film
MOVIE (see E. Y. Gong, ext. 2-7214 for this option).

The DD80 and UX80 files may be kept on disk by typing the option "K" when the output media is requested.

```
User  :  CRT
TTY   :  TMDS #, RJET #, DD80,UX80,FR80,C16,C35 or MOVIE

User  :  TMDS 660 RJET 25 K DD80
TTY   :  Type plot option : PLOTVAR VELPLOT CONTOUR OR END

User  :  PLOTVAR
TTY   :  TYPE : VARIABLE 'X' XYMIN XYMAX INDEX OR END

User  :  P X 0. 40. 2
TMDS  :  - NICE PICTURE OF PRESSURE VARIATION
TTY   :  ?
```

NOTES: The legal variables are P, U, V or VF for the pressure, X-velocity, Y-velocity, and volume fraction, respectively. The example above will plot P(pressure) in X-direction ranging from X = 0. to 40. for row J = 2. For the Y-direction, the index would represent column 1 = 2. The '?' indicates you are in the PLOTVAR mode. Typing 'END' will exit this mode and go back to the plot option level.

Typing "ENDALL" at any time will end all postprocessing and result in the message "ALL DONE".

```
User  :  END
TTY   :  Type plot option :  PLOTVAR VELPLOT CONTOUR OR END

User  :  VELPLOT
TTY   :  Type:  XMIN XMAX YMIN YMAX UMAX OR END

User  :  0.  24.  0.  40.  0.

TMDS  :  - NICE VELOCITY PLOT
TTY   :  ?
```

NOTES: UMAX is a value used to scale the velocity vector. A value of 0. will cause the code to determine the maximum velocity and scale accordingly. By entering only a linefeed (for XMIN,...,UMAX), DEFAULT, values of XMIN, XMAX, YMIN, YMAX and UMAX will be used. The '?' indicates you are in the VELPLOT mode. Typing an 'END' will exit this mode and go back to the plot option level.

If the alphanumeric character 'S' is typed instead of the UMAX value, then the finite element structure will be plotted instead of the velocity vectors.

Marker particles will be plotted in the same frame if (VELPLOT*) is entered in place of (VELPLOT).

The grid will be plotted in the same frame if (VELPLOT$) is entered in place of (VELPLOT). Velocity vectors are plotted from the cell center using values interpolated from the cell sides.

```
User  :  END
TTY   :  Type plot option : PLOTVAR VELPLOT CONTOUR OR END

User  :  CONTOUR
TTY   :  Type : XMIN XMAX YMIN YMAX VARIABLE OR END

User  :  0.  24.  0.  40.  P
TMDS  :  - NICE PICTURE OF PRESSURE CONTOUR
TTY   :  ?
User  :  VF
TMDS  :  - NICE PICTURE OF VOLUME-FRACTION CONTOUR
TTY   :  ?
```

The legal variables are P, U, V or VF. If only the variable is entered, default values of XMIN, XMAX, YMIN and YMAX are obtained from the dump. The '?' indicates you are in the contour mode. Typing an 'END' will exist this mode and go back to the plot option level.

The contour region may be shaded with different intensities by typing P%, U%, or V% instead of P, U, V.

To obtain labeled contour lines type P+, U+, or V+. Marker particles will be plotted in the same frame if (CONTOURS*) is entered in place of (CONTOUR).

The grid will be plotted in the same frame in (CONTOURS) is entered in place of (CONTOUR).

```
User  :  END
TTY   :  Type plot option : PLOTVAR VELPLOT CONTOUR OR END

User  :  END
TTY   :  Type EDIT, PRINT, CRT OR END

User  :  END
TTY   :  Type RESTART DUMP NAME AND BOX * OR END

User  :  END
TTY   :  ALL DONE
```

B. Running from a Disk File

The input options for POSTIC may be submitted through a disk input file which must be named CARDIN. Each line in the file should correspond to the input line you would type in the mode after initiating:

```
User  :  POSTIC PICXXX / t v
```

For example, the following CARDIN file will produce a DD80 file containing a velocity and pressure contour plot.

```
CRT
K KK80
V
END
C P
ENDALL
```

5.4 READING THE PRINT OUT

The first page of the print out gives an echo print of all the control cards used to generate the problem. This provides the user a permanent record of his input deck.

The title page gives the date of the controllee being used and the time and date of the run. By referring to the code development log, one can then determine what changes have been made to the code between versions.

The complete set of values in the control arrays IZ(50) and ZZ(50) is printed out so that the user can check all the default values as well as those inputted by control cards. These numbers should be checked to verify that the problem is being run as desired.

Specialized pages of the print out are described below:

Grid Coordinates

The grid coordinate variables have the following interpretation:

XC,YC	(X,Y) coordinates of cell center
XG(I)	X-coordinate of the grid line on the right side of the cell
YG(I)	Y-coordinate of the grid line on the top side of the cell
DX,DY	size of the cell.

The tags JB and JT give the sweep coverage of the grid in the Y-direction by specifying, the top and bottom grid lines in the sweep.

Flexible Structure Input

Gives all the specified control values for the structure and the initial coordinates of each node from the arrays RSH(NP) and ZSH(NP).

This page gives the contents of the structure nodal array, CSTR. For each node, L, we give

X-coordinate

Y-coordinate

X-component of velocity

Y-component of velocity

DX,DY the current displacements from the initial location.

For the elements, identified by the leading node, we give the applied pressure and the structural stresses.

Fluid Structure Interface

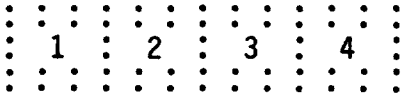
This page gives the contents of the intercept array BCSTR. The I and J line of each cell with a structural intercept along with the associated structural element are identified. The tag designates whether there are one or two intercepts in the cell and the I or J designates which intercept is used for applying the boundary conditions. The fractional intercept measured in the positive direction along the indicated I or J line is given along with the tangent of the element's slope angle at this intersection. The velocities given are those of the structure at the intercept.

Logic Grid Void Fraction Map

This gives a quick look at the range of void fractions for the entire grid. The numbers give the fraction in steps of 0.1 and shows which cells are tagged as outside the problem.

Cell Identifiers

This page gives all the cell identifiers MA1-MA10 in four blocks consolidated into a single ten digit word in the form



BLOCK 1

Contains the information in MA10. An asterisk (*) is used if the cell is tagged as outside the problem.

BLOCK 2

Contains the information in MA9 which identifies either the I or J line as the coupling line for the fluid-structure coupling.

BLOCK 3

Contains the values for MA5-MA8 giving the fluid interface fractions for the four sides numbered from right to left.

BLOCK 4

Contains the values for MA1-MA4 giving the cell side status for the four sides numbered from right to left. The letters have the meaning:

R = rigid side
L = liquid side
M = mixed side
V = void side.

If a side has a mixed tag, M, then block 3 should also give the intercept value for that side.

Surfaces

This page gives information on the top and bottom surfaces for each I value. The location, cell J number, and the applied pressure are given. The surface heights are specified at the cell center. The indices JT and JB give the J value of the cell containing the surface.

For vent clearing problems, the initial flow in the pipe is calculated using the surface algorithm. After vent clearing, the bubble algorithm is used. This allows us to use two different boundary conditions. The tag, HT, for the top surface specifies which subroutine is being used.

HT = 0 using subroutine SURF
HT = 1 using subroutine BUBBLE

Basic Variables

A separate page is devoted to the following variables for each cell

void fraction
X-velocity
Y-velocity
divergence $\nabla \cdot V$
DELP/P

The page "DEPL/P" gives the relative change of pressure in that cell that is required to make the divergence equal zero. It gives an indication as to how well the pressure was determined.

ACKNOWLEDGMENTS

This work was performed under the auspices of the U.S. Department of Energy by the Lawrence Livermore Laboratory under contract number W-7405-Eng-48. The work was sponsored by the Nuclear Regulatory Commission, Office of Nuclear Regulatory Research and is designated Task No. 189, A0116-6. Mr. Robert Cudlin is the Nuclear Regulatory Commission Program Manager.

REFERENCES

1. C. W. Hirt, B. D. Nichlos, and N. C. Romero, SOLA - A Numerical Solution Algorithm for Transient Fluid Flows, Los Alamos Scientific Laboratory, LA-5652 (1975).
2. G. L. Goudreau, A Computer Module for One Step Dynamic Response of an Axisymmetric or Plane Linear Elastic Thin Shell, Lawrence Livermore Laboratory, Rept. UCID-17730 (1978).
3. W. Kowalchuk and A. Sonin, A Model for Condensation Oscillations in a Vertical Pipe Discharging Steam into a Subcooled Water Pool, Massachusetts Institute of Technology, Report for NRC (1978).
4. W. H. McMaster, D. M. Norris, G. L. Goudreau, D. R. Quinones, E. Y. Gong, B. Moran, and N. Macken, A Coupled Fluid-Structure Method for Pressure Suppression Analysis, Lawrence Livermore Laboratory, Rept. UCRL-52620 (1979).
5. W. F. Ames, Numerical Methods for Partial Differential Equations (Barnes and Noble Publishing, New York, 1969).